SOYUT(ABSTRACT) SINIFLAR METODLAR

Öğr.Gör. Dr. Murat ASLANYÜREK

SOYUT SINIFLAR VE METODLARA GİRİŞ

- Soyut sınıflarda amaç, nesne türetilirken şablon oluşturmaktır. Soyut sınıfta tanımlanan şablon, bu sınıfı miras alan alt sınıflarda override edilerek yeniden tanımlanır.
  - Örneğin, üçgen, yamuk ve daire; geometrik şekil olması ortak özellik ama alan hesaplamaları farklıdır. Bunlar için üst sınıf ve alt sınıf ortak olmayan metod abstract(soyut) tanımlanır ve alt sınıfta soyut sınıf override edilir.
  - Soyut sınıf tanımlandığında içerisinde mutlaka bir soyut metod bulundurulması gerekmektedir.
  - Bir sınıfı veya metodu soyut olarak tanımlamak için erişim belirleyicisinden sonra abstract anahtar sözcüğünü kullanmamız gerekir.

SOYUT SINIFLAR VE METODLARA GİRİŞ

- Soyut metodlar kendi başlarına bir anlam ifade etmezler.
- **Not**: *Bir metod soyut olarak tanımlandıysa, o metodun olduğu sınıfta mutlaka soyut olarak tanımlanmalıdır.*
- **Not**: *Soyut sınıflardaki soyut sınıf alt sınıflarda override edilmezse, derleme anında hata ile karşılaşırız.*

SOYUT SINIF VE KALITIM ARASINDAKİ İLİŞKİ

- Kalıtımda olduğu gibi soyut sınıflarda bu sınıftan bir sınıf türetiliyor ve bazı gerekli metodlar override ediliyor.
- Kalıtım konusunda alt sınıftayken, üst sınıfta bulunan istediğimiz metodu override edebiliyorduk. Soyut sınıflarda soyut olarak tanımladığımız metod, alt sınıflarda mutlaka override edilmelidir. Kalıtımdaki gibi isteğe bağlı bir durum yok.
- Soyut sınıflardan, soyut alt sınıflar türetilebilir. Bu şekilde türetirsek bu alt soyut sınıf, üst soyut sınıfın soyut metodunu override etmek zorunda kalmaz.
- Bir sınıfı soyut olarak tanımlayıp, içerisinde de soyut bir metod oluşturursak, bu metodun o üst sınıf için bir anlamı olmaz. Sadece soyut sınıftan türeyen sınıflar, bu metodu kullanarak kendileri için şekillendirir.
- Soyut sınıfları kullanabilmemiz için kalıtım yapmamız gerekir.

```java
package SoyutSiniflarMetodlar;

public abstract class geometrikSekil {
    private String isim;
    public void isimBelirle(String isim){
        this.isim=isim;
    }
public String isimGetir(){
    return this.isim;
}
public abstract double alanHesap();  // soyut metod

}
```

```java
package SoyutSiniflarMetodlar;

public class Ucgen extends geometrikSekil{
    private double yukseklik;
    private double taban;
    public void bilgi(double yukseklik, double taban){
        isimBelirle("Ucgen Nesnesi");
        this.yukseklik=yukseklik;
        this.taban=taban;
    }
        @Override
        public double alanHesap(){
            return (taban*yukseklik)/2;
        }
    }
```

```java
package SoyutSiniflarMetodlar;

public class Dikdortgen extends geometrikSekil {
    private double uzunKenar;
    private double kisaKenar;

    public void bilgi(double uzunKenar, double kisaKenar){
        isimBelirle("Dikdortgen Nesnesi");
        this.uzunKenar=uzunKenar;
        this.kisaKenar=kisaKenar;
    }
        @Override
        public double alanHesap(){
            return uzunKenar*kisaKenar;
        }
    }
```

```java
package SoyutSiniflarMetodlar;

public class daire extends geometrikSekil{
    private double yaricap;
    private double pi = Math.PI;

    public void bilgi (double yaricap){
        isimBelirle("Daire Nesnesi");
        this.yaricap=yaricap;
    }
    @Override
    public double alanHesap(){
        return pi*Math.sqrt(yaricap);
    }
}
```

- Bu örneklerde soyut sınıfımızı miras aldık ve soyut sınıfımızda tanımladığımız soyut metodumuzu, bu sınıfların herbirinde override ettik.
  - Ana sınıfta soyut sınıftan bir nesne oluşturulmak istendiğinde hatayla karşılaşırız.

- **Not**: Soyut sınıflardan nesne üretilmez ama alt sınıflardan bu soyut sınıfa referans verilebilir. Ayrıca soyut sınıflar içerisinde yapıcılar tanımlanabilir.

```java
package SoyutSiniflarMetodlar;
public class soyutSinifOrnegi  {
    public static void main (String[] args)   {
        daire daireNesnesi = new daire () ;
        Dikdortgen DikdorgenNesnesi = new Dikdortgen () ;
        Ucgen UcgenNesnesi = new Ucgen () ;

        daireNesnesi.bilgi (5.0) ;
        System.out.println (daireNesnesi.isimGetir () +": ") ;
        System.out.println (daireNesnesi.alanHesap () ) ;

        DikdorgenNesnesi.bilgi (5.0, 3.0) ;
        System.out.println (DikdorgenNesnesi.isimGetir () +": ") ;
        System.out.println (DikdorgenNesnesi.alanHesap () ) ;

        UcgenNesnesi.bilgi (4.0, 3.0) ;
        System.out.println (UcgenNesnesi.isimGetir () +": ") ;
        System.out.println (UcgenNesnesi.alanHesap () ) ;
        }
}
```

- Örneklerden anlaşılacağı gibi geometrik sekil nesneleri tanımlandı ve alan hesapları yapıldı. alanHesap() metodları sınıflar içerisnde override edildiği için sonuç bütün sınıflar için özel hale geldi.

- **Not**: Soyut sınıfları farklı veriler alıp aynı işlemi yapacağımız durumlarda kullanmamız gerekmektedir.